UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/771,761	01/29/2001	Jeff A. Zimniewicz	MS160268.1	8645

27195 7590 04/25/2005

AMIN & TUROCY, LLP
24TH FLOOR, NATIONAL CITY CENTER
1900 EAST NINTH STREET
CLEVELAND, OH 44114

EXAMINER	
YIGDALL, MICHAEL J	
ART UNIT	**PAPER NUMBER**
2192	

DATE MAILED: 04/25/2005

Please find below and/or attached an Office communication concerning this application or proceeding.

UNITED STATES PATENT AND TRADEMARK OFFICE

Commissioner for Patents
United States Patent and Trademark Office
P.O. Box 1450
Alexandria, VA 22313-1450
www.uspto.gov

BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

Application Number: 09/771,761
Filing Date: January 29, 2001
Appellant(s): ZIMNIEWICZ ET AL.

Himanshu S. Amin
For Appellant

EXAMINER'S ANSWER

This is in response to the appeal brief filed February 21, 2005.

(1) Real Party in Interest

A statement identifying the real party in interest is contained in the brief.

(2) Related Appeals and Interferences

A statement identifying the related appeals and interferences which will directly affect or be directly affected by or have a bearing on the decision in the pending appeal is contained in the brief.

(3) Status of Claims

The statement of the status of the claims contained in the brief is correct.

(4) Status of Amendments After Final

The appellant's statement of the status of amendments after final rejection contained in the brief is correct.

(5) Summary of Invention

The summary of invention contained in the brief is correct.

(6) Issues

The appellant's statement of the issues in the brief is correct.

(7) Grouping of Claims

The rejection of claims 1, 8-13, 16, 18, 21-22 and 25-26 stand or fall together because Appellant's brief does not include a statement that this grouping of claims does not stand or fall together and reasons in support thereof. See 37 CFR 1.192(c)(7).

The rejection of claims 2-4, 14-15, 17, 23-24 and 29-31 stand or fall together because Appellant's brief does not include a statement that this grouping of claims does not stand or fall together and reasons in support thereof. See 37 CFR 1.192(c)(7).

The rejection of claims 5-7, 19-20 and 27-28 stand or fall together because Appellant's brief does not include a statement that this grouping of claims does not stand or fall together and reasons in support thereof. See 37 CFR 1.192(c)(7).

(8) Claims Appealed

The copy of the appealed claims contained in the Appendix to the brief is correct.

(9) Prior Art of Record

6,442,754	Curtis	8-2002
5,721,824	Taylor	2-1998
6,367,075	Kruger et al.	4-2002

(10) Grounds of Rejection

The following ground(s) of rejection are applicable to the appealed claims:

Claims 1, 8-13, 16, 18, 21-22 and 25-26 stand finally rejected under 35 U.S.C. 102(e) as being anticipated by U.S. Pat. No. 6,442,754 to Curtis ("Curtis").

Claims 2-4, 14-15, 17, 23-24 and 29-31 stand finally rejected under 35 U.S.C. 103(a) as being unpatentable over Curtis in view of U.S. Pat. No. 5,721,824 to Taylor ("Taylor").

Claims 5-7, 19-20 and 27-28 stand finally rejected under 35 U.S.C. 103(a) as being unpatentable over Curtis in view of U.S. Pat. No. 6,367,075 to Kruger et al. ("Kruger").

These rejections are set forth in a prior Office Action, mailed on September 23, 2004.

(11) Response to Argument

"Independent Claim 25" (Appellant's brief, page 4, item E):

It should be noted that this is the first time Appellant has raised the issue of 35 U.S.C. 112, sixth paragraph. Although the brief cites "36 U.S.C. §112 ¶6" (page 4, item E, second paragraph), the "36" was considered a minor typographical error.

Appellant merely asserts that claim 25 should invoke the provisions of 35 U.S.C. 112, sixth paragraph because it includes "means for" language. However, Appellant does not provide an analysis of how the language of the claim should be interpreted. The claim recites a "means for providing a valid order," a "means for controlling installation" based on the valid order, and a "means for manipulating at least one property" based on the installation. Thus, at best, any analysis of claim 25 under 35 U.S.C. 112, sixth paragraph would concentrate on the particular functions or acts of "providing," "controlling" and "manipulating," and not on the content of what is provided by the means, as primarily argued by Appellant. It should also be noted that in support of the issue of 35 U.S.C. 112, sixth paragraph, Appellant cites paragraphs [0009] and [0010] of the specification (page 4, item E, first paragraph). In paragraphs [0009] and [0010] of the specification, however, the examiner could not discern any particular means and/or explanation made as to how each of these acts should be performed.

Moreover, as noted above, Appellant's arguments are primarily directed to what is provided, rather than to how something is provided by the "means for providing" (page 6, third paragraph to page 7, second paragraph). In other words, Appellant argues for what should have been considered "a valid order." Here, the plain language of the claim is merely "a valid order," which certainly should not invoke any such "means for" language to be considered under the provisions of 35 U.S.C. 112, sixth paragraph.

"Rejection of Claims 1, 8-13, 16, 18, 21-22 and 25-26 Under 35 U.S.C. §102(e)" (pages 5-7):

Appellant contends that Curtis fails to disclose a validation engine operative to provide a valid order for the installation and/or removal of components, as claimed (page 6, third paragraph, lines 2-4). It should be noted that claim 1, for example, recites "a validation engine operative to provide a valid order" and "an installer operative to control at least one of an install and removal operation of the components based on the valid order," as emphasized by Appellant (page 6, first paragraph). It should also be noted that the function and/or act of the validation engine operative to provide a valid order should have been considered or interpreted here, rather than the semantics of "to provide."

Curtis discloses a "check_dependency" function that provides a list of components, called "dependent components," that must be installed before another component, a "depending program," is subsequently installed (see, for example, column 12, lines 27-32). In other words, the check_dependency function of Curtis provides a valid order for the installation of components. Specifically, the valid order demands that dependent components are installed first and that depending programs are installed second. Therefore, the check_dependency function

and/or act is a validation engine operative to provide a valid order. Curtis also discloses an install program that installs the components based on the valid order, which is to say that the install program uses the check_dependency function and/or act to ensure that the dependent components are installed first (see, for example, column 12, lines 32-40) and the depending program is installed second (lines 45-50). Therefore, Curtis discloses a validation engine operative to provide a valid order, and an installer operative to control an install operation of the components based on the valid order.

Appellant further contends that Curtis does not "impose" a valid ordering on the list of dependency objects created by the check_dependency function (page 6, third paragraph, lines 9-10), and that, in the case where several dependent objects are required, no order is "established" as to which of the dependent objects should be installed first (lines 13-15). Similarly, Appellant contends that, unlike the claimed invention, Curtis does not "impose" a valid order to the generated list of dependent objects (page 7, second paragraph, lines 8-10).

However, it should be noted that terms such as "impose" and "establish" are not recited in the claims. The claims merely call for "providing." Nonetheless, the valid order "imposed" by Curtis is that the dependent components must be installed before the depending program is installed (see, for example, column 12, lines 27-32), as noted above. Therefore, notwithstanding Appellant's characterization, Curtis does disclose providing a valid order and installing the components based on the valid order, as presented above. There is no need for Curtis to disclose and/or specify a particular, relative order between each dependent component to anticipate the plain language of the claims.

Appellant is reminded that although the claims are interpreted in light of the specification, limitations from the specification are not read into the claims. See *In re Van Geuns*, 988 F.2d 1181, 26 USPQ2d 1057 (Fed. Cir. 1993). For example, although Appellant contends that "the claimed invention resolves and imposes an ordering for each dependent component, thus establishing the particular order in which the particular component should be installed/removed prior to the installation of the depending component" (page 7, first paragraph, lines 2-4), claim 1 merely calls for providing a valid order and installing or removing the components based on the valid order. Much like "impose" and "establish," as noted above, terms such as "resolve" are not recited in the claims.

Furthermore, Appellant appears to suggest that Curtis fails to disclose that the installer manipulates at least one property associated with at least one shared component to reflect dependency for the at least one shared component according to the installation or removal of that component (page 6, second paragraph, lines 5-8).

However, this limitation was addressed in the claim rejections, and Appellant has not pointed out how the language of the claims patentably distinguishes the claims from Curtis, as required by 37 CFR 1.111(b) and (c).

"Rejection of Claims 2-4, 14-15, 17, 23-24 and 29-31 Under 35 U.S.C. §103(a)" (pages 7-9):

Appellant contends that Taylor fails to "establish" and "impose" an ordering on the installation and/or removal of components (page 8, last paragraph, lines 5-6). Appellant further contends that both Curtis and Taylor fail to teach or suggest a validation component that provides a valid ordering of components that need to be installed/removed (page 9, first

paragraph, lines 1-3), and that Taylor fails to initiate installation of each of the components according to the valid order "established" by the validation component (lines 3-5).

However, it should be noted that Taylor is not relied upon to introduce these features to Curtis. The check_dependency function of Curtis is a validation component that provides a valid order, and Curtis discloses installing or initiating the installation of the components based on the valid order, as presented above. Again, as noted above, Appellant argues for unclaimed merits of distinction such as "establish" and "impose," and thus the argument is moot and not persuasive. Nonetheless, Curtis here at least "imposes" the valid order on the installation in that the dependent components are installed first and the depending programs are installed second, also as presented above.

"Rejection of Claims 5-7, 19-20 and 27-28 Under 35 U.S.C. §103(a)" (page 9):

Appellant contends that Kruger does not make up for the aforementioned deficiencies with respect to Curtis (page 9, last paragraph). The deficiencies alleged by Appellant with respect to Curtis have been addressed above.

The claim rejections set forth in the Office Action mailed on September 23, 2004 are reproduced below for completeness:

Claim Rejections - 35 USC § 102

6. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (e) the invention was described in (1) an application for patent, published under section 122(b), by another filed in the United States before the invention by the applicant for patent or (2) a patent granted on an application for patent by another filed in the United States before the invention by the applicant for patent, except that an international application filed under the treaty defined in section 351(a) shall have the effects for purposes of this subsection of an application filed in the United States only if the international application designated the United States and was published under Article 21(2) of such treaty in the English language.

7. Claims 1, 8-13, 16, 18, 21, 22, 25 and 26 are rejected under 35 U.S.C. 102(e) as being anticipated by U.S. Pat. No. 6,442,754 to Curtis.

With respect to claim 1 (original), Curtis discloses a system to facilitate installation and/or removal of components (see the title and abstract) including at least one shared component (see column 4, lines 33-37, which shows components that are depended upon by more than one program, i.e. shared components), comprising:

(a) a validation engine operative to provide a valid order (see column 11, lines 11-20, which shows a function for checking dependencies, i.e. a validation engine; see also column 12, lines 22-32, which shows generating a list of dependent components and providing a valid installation order); and

(b) an installer operative to control at least one of an install and removal operation of the components based on the valid order and operative to effect manipulation of at least one property associated with the at least one shared component to reflect dependency for the at least one shared component according to the installation or removal thereof (see column 12, lines 32-50, which shows an installer for installing the components based on the valid order; see also FIG. 5 and column 13, lines 7-10, which

shows a data structure having properties that reflect dependency; see also column 13, lines 28-29, which shows manipulating the data structure when a component is installed).

With respect to claim 8 (currently amended), Curtis further discloses the limitation wherein the at least one property further comprises configuration data indicative of an operating relationship of the at least one shared component and each installed dependent component associated with the at least one shared component (see FIG. 5 and column 13, lines 7-27, which shows a data structure having properties indicative of the relationship between a component and its dependencies).

With respect to claim 9 (original), Curtis further discloses the limitation wherein the installer is operative to control installation of the at least one shared component, such that a single set of files for the at least one shared component is copied as part of the installation for use by associated dependent components (see column 9, lines 47-64, which shows determining whether dependencies are already installed and installing a set of files for a shared component).

With respect to claim 10 (original), Curtis further discloses the limitation wherein the at least one shared component has associated metadata operable to identify the at least one shared component as a shared component (see FIG. 3 and column 9, lines 10-25, which shows a dependency object comprising metadata that identifies whether a component is a shared component).

With respect to claim 11 (original), Curtis further discloses the limitation wherein the at least one shared component requires at least one dependent component to perform a substantially useful function (see column 9, lines 25-31, which shows that dependent components must be installed in order for another component to perform all intended functions).

With respect to claim 12 (original), Curtis further discloses the limitation wherein a runtime dependency exists between an installed dependent component and the shared component on which the dependent component depends (see column 9, lines 39-43, which shows dependencies needed by a component in order to operate, i.e. dependencies needed at runtime).

With respect to claim 13 (original), Curtis further discloses a system to facilitate installation of components (see the title and abstract) including at least one shared component (see column 4, lines 33-37, which shows components that are depended upon by more than one program, i.e. shared components), comprising:

(a) a setup manager which controls installation of the components (see column 5, lines 56-60, which shows an installer script, i.e. a setup manager);

(b) dependency manager which provides a valid installation order based on metadata associated with at least some of the components (see column 11, lines 11-20, which shows a function for checking dependencies, i.e. a dependency manager, using dependency objects; see also FIG. 3 and column 9, lines 10-25, which shows that the

dependency objects comprise metadata; see also column 12, lines 22-32, which shows generating a list of dependent components and providing a valid installation order);

wherein the setup manager causes the components to be installed according to the valid installation order, a separate shared installation of the at least one shared component being implemented for each dependent component that depends on the at least one shared component (see column 12, lines 32-50, which shows installing each component based on the valid installation order).

With respect to claim 16 (original), see the explanation for claim 10 above.

With respect to claim 18 (original), Curtis further discloses at least one property associated with an installed instance of the at least one shared component which reflects dependency for the at least one shared component (see FIG. 5 and column 13, lines 7-10, which shows a data structure having properties that reflect dependency).

With respect to claim 21 (original), see the explanation for claim 8 above.

With respect to claim 22 (currently amended), see the explanation for claim 9 above.

With respect to claim 25 (original), Curtis discloses a system to facilitate installation and/or removal of components (see the title and abstract) including at least one shared component (see column 4, lines 33-37, which shows components that are depended upon by more than one program, i.e. shared components), comprising:

(a) means for providing a valid order for the components (see column 12, lines 22-32, which shows generating a list of dependent components and providing a valid installation order);

(b) means for controlling installation of the components based on the valid order (see column 12, lines 32-50, which shows an installer for installing the components based on the valid order); and

(c) means for manipulating at least one property associated with the at least one shared component to reflect dependency for the at least one shared component based on at least one installation of the shared component and removal of a dependent component that depends on the at least one shared component (see FIG. 5 and column 13, lines 7-10, which shows a data structure having properties that reflect dependency; see also column 13, lines 28-29, which shows manipulating the data structure when a component is installed).

With respect to claim 26 (original), Curtis discloses a method to facilitate installing and/or removing components (see the title and abstract) including at least one shared component (see column 4, lines 33-37, which shows components that are depended upon by more than one program, i.e. shared components), the method comprising:

(a) providing a valid order (see column 12, lines 22-32, which shows generating a list of dependent components and providing a valid installation order);

(b) installing each of the plurality of components based on the valid order (see column 12, lines 32-50, which shows installing the components based on the valid order); and

(c) modifying at least one property associated with the at least one shared component to reflect dependency characteristics of the at least one shared component relative dependent components operative to use the at least one shared component (see Fig. 5 and column 13, lines 7-10, which shows a data structure having properties that reflect dependency; see also column 13, lines 28-29, which shows manipulating the data structure when a component is installed).

Claim Rejections - 35 USC § 103

8. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

9. Claim 2-4, 14, 15, 17, 23, 24 and 29-31 are rejected under 35 U.S.C. 103(a) as being unpatentable over Curtis as applied to claims 1, 13 and 26 above, respectively, in view of U.S. Pat. No. 5,721,824 to Taylor.

With respect to claim 2 (original), although Curtis shows that shared, or dependent, components are identified for installation prior to non-shared components (see column 12, lines 27-32), Curtis does not expressly disclose the limitation wherein the

valid order identifies shared components for installation subsequent to non-shared components.

However, Taylor discloses the limitation above in terms of an action list, i.e. a valid order, that identifies shared, or dependent, components for installation subsequent to a non-shared package (see column 5, lines 25-29). Note that Taylor also discloses an implementation wherein dependent components are identified for installation before non-shared components, as in the Curtis system (see column 7, lines 49-53).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to modify the Curtis system with the feature of identifying shared components for installation subsequent to non-shared components, as taught by Taylor, for the purpose of supporting an installation sequence that conforms to the constraints of the target system (see Taylor, column 2, lines 1-3), in order to increase the compatibility of the installation routine with different platforms.

With respect to claim 3 (original), Curtis further discloses the limitation wherein the installer is operative to initiate a method to install each of the components based on the valid order during a part of the installation (see column 12, lines 32-50, which shows an installer for installing each of the components based on the valid order; see also column 12, lines 59-62, which shows that the installer is operative to initiate the installation).

Curtis does not expressly disclose the limitation wherein the at least one shared component is installed and configured for a selected dependent component during the first part of installation.

However, Taylor further discloses the limitation above in terms of installing components based on the action list, i.e. the valid order, during a first part of the installation (see column 2, lines 7-11, which shows that the flow of operations is layered, i.e. has multiple parts; see also column 2, lines 12-26, which shows installing packages during a first part of the installation).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to modify the Curtis system with the feature of installing components during a first part of the installation, as taught by Taylor, for the purpose of installing multiple software packages with a single load on the system, in order to improve the perceived performance (see Taylor, column 3, lines 47-51).

With respect to claim 4 (original), Curtis does not disclose the limitation wherein the method is operative to employ a second part of the installation to install the at least one shared component for each dependent component other than the selected dependent component during the second part of the installation.

Taylor further discloses the limitation above in terms of installing packages or components that are depended upon by other dependent packages during a second part of the installation (see column 2, lines 7-11, which shows that the flow of operations is layered, i.e. has multiple parts; see also column 2, lines 53-62, which shows installing packages during a second part of the installation).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to modify the Curtis system with the feature of installing components during a second part of the installation, as taught by Taylor, for the purpose

of installing multiple software packages with a single load on the system, in order to improve the perceived performance (see Taylor, column 3, lines 47-51).

With respect to claim 14 (original), although Curtis does show a dependency manager for generating a valid installation order (see column 11, lines 11-20, which shows a function for checking dependencies, i.e. a dependency manager; see also column 12, lines 22-32, which shows generating a list of dependent components and providing a valid installation order), Curtis does not expressly disclose the limitation wherein the dependency manager is operative to validate a received installation order, which, upon validation of the received installation order, becomes the valid installation order.

However, Taylor further discloses the limitation above in terms of validating a dependency list, i.e. a received installation order, and using it as the valid installation order (see column 2, lines 28-40, which shows translating the dependency list into an action list, i.e. a valid installation order).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to modify the Curtis system with the feature of validating an installation order, as taught by Taylor, for the purpose of installing multiple software packages with a single load on the system, in order to improve the perceived performance (see Taylor, column 3, lines 47-51).

With respect to claim 15 (original), although Curtis does show a dependency manager for generating a valid installation order (see column 11, lines 11-20, which shows a function for checking dependencies, i.e. a dependency manager; see also column

12, lines 22-32, which shows generating a list of dependent components and providing a valid installation order), Curtis does not expressly disclose the limitation wherein, if the received installation order is improper, the dependency manager is operative to create the valid installation order.

However, Taylor further discloses the limitation above in terms of validating a dependency list, i.e. a received installation order, and creating a valid installation order (see column 2, lines 12-26, which shows generating an action list, i.e. a valid installation order).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to modify the Curtis system with the feature of validating an installation order, as taught by Taylor, for the purpose of installing multiple software packages with a single load on the system, in order to improve the perceived performance (see Taylor, column 3, lines 47-51).

With respect to claim 17 (original), although Curtis does show an installer or setup manager for installing each of the components based on the valid installation order (see column 12, lines 32-50; see also column 12, lines 59-62, which shows that the installer is operative to initiate the installation), Curtis does not expressly disclose the limitation wherein the setup manager is operative to initiate a method to install each of the components according to the valid installation order during a first part of the installation, the at least one shared component being installed for a first dependent component during the first part of installation, the method being operative to install the at

least one shared component for each other dependent component during a second part of the installation.

However, Taylor further discloses the limitation above in terms of installing components based on the action list, i.e. the valid installation order, during a first part of the installation, and installing components that are depended upon by other dependent packages during a second part of the installation (see column 2, lines 7-11, which shows that the flow of operations is layered, i.e. has multiple parts; see also column 2, lines 12-26, which shows installing packages during a first part of the installation; see also column 2, lines 53-62, which shows installing packages during a second part of the installation).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to modify the Curtis system with the feature of first and second installation parts, as taught by Taylor, for the purpose of installing multiple software packages with a single load on the system, in order to improve the perceived performance (see Taylor, column 3, lines 47-51).

With respect to claim 23 (original), Curtis discloses a system to facilitate installation and/or removal of components (see the title and abstract) including at least one shared component (see column 4, lines 33-37, which shows components that are depended upon by more than one program, i.e. shared components), comprising:

(a) a validation component operative to provide a valid order based on setup data (see column 11, lines 11-20, which shows a function for checking dependencies, i.e. a validation component, using dependency objects; see also FIG. 3 and column 9, lines 10-25, which shows that the dependency objects comprise setup data; see also column 12,

lines 22-32, which shows generating a list of dependent components and providing a valid installation order).

Although Curtis does show an installer or setup engine for installing each of the components based on the valid installation order (see column 12, lines 32-50; see also column 12, lines 59-62, which shows that the installer is operative to initiate the installation), Curtis does not expressly disclose the limitation of:

(b) a setup engine operative to initiate installation of each of the components according to the valid order during a first part of the installation, the shared component being installed for a first dependent component during the first part of installation, the shared component being installed for each other dependent component during a second part of the installation separate from the first part.

However, Taylor discloses the limitation above in terms of installing components based on the action list, i.e. the valid installation order, during a first part of the installation, and installing components that are depended upon by other dependent packages during a second part of the installation (see column 2, lines 7-11, which shows that the flow of operations is layered, i.e. has multiple parts; see also column 2, lines 12-26, which shows installing packages during a first part of the installation; see also column 2, lines 53-62, which shows installing packages during a second part of the installation).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to modify the Curtis system with the feature of first and second installation parts, as taught by Taylor, for the purpose of installing multiple software

packages with a single load on the system, in order to improve the perceived performance (see Taylor, column 3, lines 47-51).

With respect to claim 24 (currently amended), Curtis discloses a system to facilitate installation and/or removal of components (see the title and abstract) including at least one shared component (see column 4, lines 33-37, which shows components that are depended upon by more than one program, i.e. shared components), comprising:

(a) a dependency manager operative to provide a valid order based on setup data (see column 11, lines 11-20, which shows a function for checking dependencies, i.e. a dependency manager, using dependency objects; see also Fig. 3 and column 9, lines 10-25, which shows that the dependency objects comprise setup data; see also column 12, lines 22-32, which shows generating a list of dependent components and providing a valid installation order).

Although Curtis does show an installer or setup engine for installing each of the components based on the valid installation order (see column 12, lines 32-50; see also column 12, lines 59-62, which shows that the installer is operative to initiate the installation), Curtis does not expressly disclose the limitation of:

(b) a setup engine operative to initiate installation of each of the components according to the valid order during a first part of the installation, the shared component being installed for a first dependent component during the first part of installation, the shared component being installed for each other dependent component during a second part of the installation, which is subsequent to the first part.

However, Taylor discloses the limitation above in terms of installing components based on the action list, i.e. the valid installation order, during a first part of the installation, and installing components that are depended upon by other dependent packages during a second part of the installation (see column 2, lines 7-11, which shows that the flow of operations is layered, i.e. has multiple parts; see also column 2, lines 12-26, which shows installing packages during a first part of the installation; see also column 2, lines 53-62, which shows installing packages during a second part of the installation).

Curtis further discloses the limitation wherein the setup manager is operative to effect manipulation of at least one property associated with the at least one shared component to reflect dependency characteristics of the at least one shared component as a function of at least one of installation of the shared component and removal of a dependent component that depends on the at least one shared component (see Fig. 5 and column 13, lines 7-10, which shows a data structure having properties that reflect dependency characteristics; see also column 13, lines 28-29, which shows manipulating the data structure when a component is installed).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to modify the Curtis system with the feature of first and second installation parts, as taught by Taylor, for the purpose of installing multiple software packages with a single load on the system, in order to improve the perceived performance (see Taylor, column 3, lines 47-51).

With respect to claim 29 (original), see the explanation for claim 3 above.

With respect to claim 30 (original), see the explanation for claim 4 above.

With respect to claim 31 (original), Curtis discloses a method to facilitate installing and/or removing components including at least one shared component, the method comprising:

(a) providing a valid order (see column 12, lines 22-32, which shows generating a list of dependent components and providing a valid installation order).

Although Curtis does show an installer for installing each of the components based on the valid order (see column 12, lines 32-50; see also column 12, lines 59-62, which shows that the installer is operative to initiate the installation), Curtis does not expressly disclose the steps of:

(b) effecting installation of each of the components during a first part of installation according to the valid order, the shared component being installed for a first dependent component during the first part of the installation;

(c) effecting installation of the shared component for each other dependent component during a second part of the installation separate from the first part.

However, Taylor discloses step (b) above in terms of installing components based on the action list, i.e. the valid order, during a first part of the installation (see column 2, lines 7-11, which shows that the flow of operations is layered, i.e. has multiple parts; see also column 2, lines 12-26, which shows installing packages during a first part of the installation).

Taylor further discloses step (c) above in terms of installing packages or components that are depended upon by other dependent packages during a second part of

the installation (see column 2, lines 7-11, which shows that the flow of operations is layered, i.e. has multiple parts; see also column 2, lines 53-62, which shows installing packages during a second part of the installation).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to modify the Curtis system with the feature of first and second installation parts, as taught by Taylor, for the purpose of installing multiple software packages with a single load on the system, in order to improve the perceived performance (see Taylor, column 3, lines 47-51).

10. Claims 5-7, 19, 20, 27 and 28 are rejected under 35 U.S.C. 103(a) as being unpatentable over Curtis as applied to claims 1, 18 and 26 above, respectively, in view of U.S. Pat. No. 6,367,075 to Kruger et al. (hereinafter "Kruger").

With respect to claim 5 (original), although Curtis does show storing dependency information in order to indicate which components depend on a shared component (see column 13, lines 1-6), Curtis does not expressly disclose the limitation wherein the at least one property further comprises a reference count having a value indicative of a number of dependent components associated with the at least one shared component.

However, Kruger discloses the limitation above in terms of an installer that uses a reference count for shared library files, to ensure that files depended upon by other programs are not affected (see column 9, lines 14-21).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to modify the Curtis system with the reference count feature taught

by Kruger et al., for the purpose of ensuring that shared components used by other programs are not affected, inherently reducing the number of potential version conflicts.

With respect to claim 6 (original), although Curtis does show storing dependency information in order to indicate which components depend on a shared component (see column 13, lines 1-6; note that the information is written during installation), Curtis does not expressly disclose the limitation wherein the installer is operative to effect an increase in the value of the reference count for each installation of the at least one shared component.

However, Kruger further discloses the limitation above in terms of incrementing the reference count when a file is added or installed, to ensure that files depended upon by other programs are not affected (see column 9, lines 14-21).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to modify the Curtis system with the reference count feature taught by Kruger et al., for the purpose of ensuring that shared components used by other programs are not affected, inherently reducing the number of potential version conflicts.

With respect to claim 7 (original), although Curtis does show storing dependency information in order to indicate which components depend on a shared component (see column 13, lines 1-6; note that the information is used when a component is to be uninstalled), Curtis does not expressly disclose the limitation wherein the installer is operative to effect a decrease in the value of the reference count in response to removal of a dependent component that depends on the at least one shared component.

However, Kruger further discloses the limitation above in terms of decreasing the reference count when a file is deleted or removed, to ensure that files depended upon by other programs are not affected (see column 9, lines 14-21).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to modify the Curtis system with the reference count feature taught by Kruger et al., for the purpose of ensuring that shared components used by other programs are not affected, inherently reducing the number of potential version conflicts.

With respect to claim 19 (original), see the explanation for claim 5 above.

With respect to claim 20 (original), see the explanations for claims 6 and 7 above.

With respect to claim 27 (original), see the explanation for claim 5 above.

With respect to claim 28 (original), see the explanations for claims 6 and 7 above.

For the above reasons, it is believed that the rejections should be sustained.

Respectfully submitted,

MY

Michael J. Yigdall
Examiner
Art Unit 2192

mjy
April 14, 2005

Conferees
Tuan Dam, SPE 2192
Kakali Chaki, SPE 2193

TUAN DAM
SUPERVISORY PATENT EXAMINER

KAKALI CHAKI
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100

AMIN & TUROCY, LLP
24TH FLOOR, NATIONAL CITY CENTER
1900 EAST NINTH STREET
CLEVELAND, OH 44114